



**HP64000  
Logic Development  
System**

**Model 64000  
Assembler Supplement  
6805/6809**



**HEWLETT  
PACKARD**

## **CERTIFICATION**

*Hewlett-Packard Company certifies that this product met its published specifications at the time of shipment from the factory. Hewlett-Packard further certifies that its calibration measurements are traceable to the United States National Bureau of Standards, to the extent allowed by the Bureau's calibration facility, and to the calibration facilities of other International Standards Organization members.*

## **WARRANTY**

This Hewlett-Packard system product is warranted against defects in materials and workmanship for a period of 90 days from date of installation. During the warranty period, HP will, at its option, either repair or replace products which prove to be defective.

Warranty service of this product will be performed at Buyer's facility at no charge within HP service travel areas. Outside HP service travel areas, warranty service will be performed at Buyer's facility only upon HP's prior agreement and Buyer shall pay HP's round trip travel expenses. In all other cases, products must be returned to a service facility designated by HP.

For products returned to HP for warranty service. Buyer shall prepay shipping charges to HP and HP shall pay shipping charges to return the product to Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to HP from another country.

HP warrants that its software and firmware designated by HP for use with an instrument will execute its programming instructions when properly installed on that instrument. HP does not warrant that the operation of the instrument, or software, or firmware will be uninterrupted or error free.

## **LIMITATION OF WARRANTY**

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by Buyer, Buyer-supplied software or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. HP SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

## **EXCLUSIVE REMEDIES**

THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES. HP SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, TORT, OR ANY OTHER LEGAL THEORY.

## **ASSISTANCE**

*Product maintenance agreements and other customer assistance agreements are available for Hewlett-Packard products.*

*For any assistance, contact your nearest Hewlett-Packard Sales and Service Office.*

FOLD HERE



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

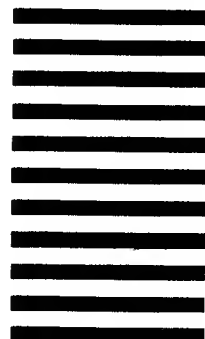
**BUSINESS REPLY CARD**

FIRST CLASS PERMIT NO. 1303 COLORADO SPRINGS, COLORADO

POSTAGE WILL BE PAID BY ADDRESSEE

**HEWLETT-PACKARD**

Logic Product Support Dept.  
Attn: Technical Publications Manager  
Centennial Annex - D2  
P.O. Box 617  
Colorado Springs, Colorado 80901-0617



FOLD HERE

Your cooperation in completing and returning this form  
will be greatly appreciated. Thank you.

# READER COMMENT SHEET

Op. Manual, Model 64000  
Assembler Supplement 6805/6809  
64844-90904, July 1983

**Part Number:** \_\_\_\_\_

Your comments are important to us. Please answer this questionnaire and return it to us. Circle the number that best describes your answer in questions 1 through 7. Thank you.

1. The information in this book is complete:

Doesn't cover enough  
(what more do you need?)

1 2 3 4 5

Covers everything

2. The information in this book is accurate:

Too many errors

1 2 3 4 5

Exactly right

3. The information in this book is easy to find:

I can't find things I need

1 2 3 4 5

I can find info quickly

4. The Index and Table of Contents are useful:

Helpful

1 2 3 4 5

Missing or inadequate

5. What about the "how-to" procedures and examples:

No help

1 2 3 4 5

Very helpful

Too many now

1 2 3 4 5

I'd like more

6. What about the writing style:

Confusing

1 2 3 4 5

Clear

7. What about organization of the book:

Poor order

1 2 3 4 5

Good order

8. What about the size of the book:

too big/small

1 2 3 4 5

Right size

Comments: \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

Particular pages with errors?

\_\_\_\_\_

Name (optional): \_\_\_\_\_

Job title: \_\_\_\_\_

Company: \_\_\_\_\_

Address: \_\_\_\_\_

**Note:** If mailed outside U.S.A., place card in envelope. Use address shown on other side of this card.

## **Printing History**

Each new edition of this manual incorporates all material updated since the previous edition. Manual change sheets are issued between editions, allowing you to correct or insert information in the current edition.

The part number on the back cover changes only when each new edition is published. Minor corrections or additions may be made as the manual is reprinted between editions.

First Printing . . . . . April 1980 (Part Number 64844-90902)  
Second Edition . . . . February 1981 (Part Number 64844-90904)  
Reprinted . . . . . May 1982  
Reprinted . . . . . July 1983



# **Assembler Supplement 6805/6809**

© COPYRIGHT HEWLETT-PACKARD COMPANY 1980, 1981, 1982, 1983  
LOGIC SYSTEMS DIVISION  
COLORADO SPRINGS, COLORADO, U.S.A.

ALL RIGHTS RESERVED

# Table of Contents

## Chapter 1: General Information

Introduction .....	1-1
MC6805 Architecture .....	1-1
General .....	1-1
Accumulator Register (A) .....	1-1
Program Counter Register (PC) .....	1-1
Index Register (X) .....	1-2
Stack Pointer Register (SP) .....	1-2
Condition Code Register (CC) .....	1-2
MC6809 Architecture .....	1-2
General .....	1-2
Accumulating Registers .....	1-2
Program Counter Register (PC) .....	1-3
Index Registers (X and Y) .....	1-3
Stack Pointer Registers (U and S) .....	1-3
Condition Code Register (CC) .....	1-3
Direct Page Register (DP) .....	1-3
Modes of Addressing .....	1-4
Inherent Addressing .....	1-4
Immediate Addressing .....	1-4
Direct and Extended Addressing .....	1-4
Relative Addressing .....	1-5
Indexed Addressing (MC6805 only) .....	1-5
Indexed Addressing (MC6809 only) .....	1-6
Indexed Indirect Addressing (MC6809 only) .....	1-10
Bit Set/Clear Addressing (MC6805 only) .....	1-11
Bit Test and Branch (MC6805 only) .....	1-11
Condition Codes .....	1-11
Carry Borrow (C) Register .....	1-12
Overflow (V) Register (MC6809 only) .....	1-12
Zero (Z) Register .....	1-12
Negative (N) Register .....	1-12
IRQ Interrupt Mask (I) Register .....	1-12
Half Carry (H) Register .....	1-12
FIRQ Interrupt Mask (F) Register (MC6809 only) .....	1-13
Entire Flag (E) Register (MC6809 only) .....	1-13

## Chapter 2: Operand Rules and Conventions

Types of Information .....	2-1
Additional Operand Information .....	2-2
Immediate Addressing Indicator .....	2-2
Indexed Addressing Indicator .....	2-3
Direct-Extended Addressing Mode Default .....	2-3
Location Counter Indicator .....	2-3
Operand Expressions .....	2-4



## Table of Contents (Cont'd)

### Chapter 3: Special Pseudo Instructions

Introduction .....	3-1
BASE_SEG, BASE_END .....	3-2
BEXT .....	3-3
BSZ .....	3-4
DIRECT EXTEND .....	3-5
FCB .....	3-6
FCC .....	3-7
FDB .....	3-8
REG .....	3-9
RMB .....	3-10
SET .....	3-11
SETDP .....	3-12

### Chapter 4: Assembler Output Listing

General .....	4-1
Input/Output Files .....	4-1
Source Input File .....	4-1
Assembler Output Files .....	4-1
Output Listing .....	4-2

### Chapter 5: Instruction Set Summary

General .....	5-1
Predefined Symbols .....	5-3

## List of Tables

1-1.	Instruction Addressing Modes - MC6805 .....	1-14
1-2.	Instruction Addressing Modes - MC6809 .....	1-15
1-3.	Indexed Addressing Modes (MC6809 only) .....	1-16
4-1.	Source Program Format Example .....	4-3
4-2.	Assembler Output Listing .....	4-4
4-3.	Assembler Output with Errors .....	4-6
5-1.	MC6805 Instruction Set Summary .....	5-4
5-2.	MC6809 Instruction Set Summary .....	5-13



# Chapter 1

## General Information (6805/6809)

### Introduction

This chapter contains general information about the 6805 microcomputer and 6809 microprocessor. It briefly discusses their architecture, addressing modes, and condition codes. For a detailed description of a specific device, refer to the manufacturer's Programming Reference Manual.

#### NOTE

---

Use the processor number for the assembler directive; i.e. "6805" or "6809"

---

#### NOTE

---

The following paragraphs apply to both the MC6805 and MC6809. Paragraphs and subparagraphs that apply to only one device will so indicate in their title.

---

### MC6805 Architecture

#### General

There are five registers available in the MC6805 microcomputer. These registers are discussed briefly in the following paragraphs.

#### Accumulator Register (A)

The microcomputer has one 8-bit register that functions as an accumulator for arithmetic calculations and data manipulations.

#### Program Counter Register (PC)

The 11-bit program counter register contains the address of the next instruction to be executed.

### **Index Register (X)**

The index register is a special-purpose 8-bit register used for the indexed addressing mode. It may also be used for limited calculations and data manipulations when using read/modify/write instructions. When not required for indexing, it may be used as a temporary storage area.

### **Stack Pointer Register (SP)**

The stack pointer is a special-purpose 11-bit register that contains the address of the next usable location on the stack. The six most significant bits of the stack pointer are permanently set to 000011B. During a microcomputer reset or the reset stack pointer (RSP) instruction, the stack pointer will be set to location 07FH. Subroutines and interrupts may be nested down to location 061H.

### **Condition Code Register (CC)**

The condition code register contains the five flags which indicate the results of the instruction just executed. These flags may be individually tested for conditional branching purposes. A description of each condition code is given later in this chapter.

## **MC6809 Architecture**

### **General**

There are nine registers available in the MC6809 microprocessor. These registers are discussed briefly in the following paragraphs.

### **Accumulating Registers**

The microprocessor has two registers that function as accumulators for arithmetic calculations and data manipulation. They are referred to as register A and register B. Each register has its own group of instructions and the mnemonic of the source statement specifies which register is to be used. For example:

ROLA - Rotate content of register A to the left.

ROLB - Rotate content of register B to the left.

CLRA - Clear register A.

CLRB - Clear register B.

In addition, certain instructions join the two registers to form a 16-bit register. When used in this configuration, the combined registers are referred to as the D register and the most significant byte of the 16-bit word is maintained in register A.

### **Program Counter Register (PC)**

The 16-bit program counter of the microprocessor may specify up to 65,536 addresses. When using the relative addressing mode of operation, the program counter register may also be used as an index register (in specific operations).

### **Index Registers (X and Y)**

The index registers are special-purpose 16-bit registers used in the indexed addressing mode of operation. The address in each register allows the microprocessor to point to memory locations directly or they may be altered to produce a register offset. During certain operations, the registers may be incremented or decremented to point to the next location in memory.

### **Stack Pointer Registers (U and S)**

The stack pointer registers are another set of special-purpose 16-bit registers and have the same indexed addressing capabilities as the index registers (X and Y).

The S (hardware) stack pointer register is used by the microprocessor during interrupt and subroutine calls.

The U (user) stack pointer register is controlled exclusively by the programmer.

#### **NOTE**

---

Both stack pointers point to the top of the stack as opposed to some microprocessors where the stack pointer points to the next vacant location on the stack.

---

### **Condition Code Register (CC)**

The microprocessor has eight condition codes that make up bits 0 through 7 of an 8-bit register. The eight condition codes and their use are discussed later in this chapter.

### **Direct Page Register (DP)**

The direct page register permits addressing directly any location in memory. The content of this register appears on the address bus (A8-A15) during the execution of instructions using the direct mode of addressing.

## Modes of Addressing

Instructions for microprocessors may be separated into a number of categories, but their most common attribute is their modes of addressing. An addressing mode refers to the method by which an instruction addresses its operand. The addressing modes for each MC6805 instruction are listed in table 1-1. The addressing modes for each MC6809 instruction are listed in table 1-2.

### Inherent Addressing

The inherent mode of addressing requires no operand and extended addressing is not permitted. All instructions that use this form of addressing are one-byte operations.

### Immediate Addressing

In this mode of addressing, the instruction contains the value of the operand to be used in the operation or computation. The only instructions permitted for this mode of addressing are indicated in table 1-1 and table 1-2 under the column labelled "Immediate."

To select this mode of addressing, the corresponding operand must be preceded by the pound (#) character. The operand data may be in the form of an ASCII character, a number, a label, or an expression. The microprocessor uses both 8- and 16-bit immediate values depending on the size specified by the opcode.

### Direct and Extended Addressing

In direct addressing, an instruction requires two bytes of memory. The first byte will be the opcode of the instruction and the second byte will be the absolute numerical address where the operand is located.

In extended addressing, the instruction uses three bytes of memory with the first byte containing the opcode of the instruction, the second byte containing the highest 8 bits of the absolute numerical address, and the third byte containing the lowest 8 bits of the absolute numerical address.

For those instructions that can use both direct and extended modes, the assembler defaults to extended for externals, relocatables, and forward references. The direct mode is used when addresses are in the 0 to FFH range. The default function can be overridden by using the DIRECT pseudo instruction. Once the direct pseudo is inserted in a source program, the direct mode of addressing will be in effect until cancelled by an EXTEND pseudo instruction.

## Relative Addressing

Branch instructions are somewhat different from other instructions in that their associated addresses do not indicate the location of data. Instead, the address indicates the location of the next instruction that is to be executed. This location is relative to the current setting of the program counter. There are two forms of branch instructions.

For the short branch relative addressing mode to be valid, the distance of the branch must fall in the value range of -126 to +129 bytes. This relationship between the relative address and the absolute address of the destination of the branch may be expressed by the following:

$$DA = (PC+2)+R$$

where:

DA = address of the destination of the branch instruction.

PC = content of the program counter.

R = the 8-bit, two's complement, binary number listed in the second byte of the instruction.

The long branch relative addressing mode may operate in the full range of addressing (0000H - FFFFH). In this mode of operation, a two-byte offset is calculated and placed in the operand field of the branch instruction. The offset is the two's complement value of the difference between the location of the byte immediately following the opcode and the location of the destination of the branch.

## Indexed Addressing (MC6805 only)

**No Offset.** This mode of operation addresses the lowest 256 bytes of memory. The instructions are one-byte operations and the index register points to the destination address.

**8-bit Offset.** The destination address will be calculated by adding the content of the byte following the opcode to the content of the index register. This mode of operation addresses the lowest 511 bytes of memory. The instructions using this mode of operation are two-byte operations.

**16-bit Offset.** This mode of operation calculates the destination address by adding the content of the two bytes following the opcode to the index register. Thus, the entire memory space may be addressed. Instructions that use this mode of operation are three-byte operations.

## Indexed Addressing (MC6809 only)

Indexed addressing relates to one of the index registers (X, Y, U, S, and sometimes PC). The address will be determined at the time of execution by adding the value specified in the operand field to the current content of the designated register. There are five modes of indexed addressing (both indirect and non-indirect) and they are given in table 1-3.

### NOTE

---

The indexed indirect mode of addressing is selected by enclosing the operand field in brackets.

---

The value of the opcode postbyte (first byte following the opcode) listed in table 1-3 is determined by the format of the operand. The two indexing formats are as follows:

- a. Simple format indexing: this type of formatting takes the form:

expr,R

where: expr is an absolute expression in the range -16 to +15 but not zero, and R designates one of the index registers X, Y, U, or S. The postbyte code for simple format indexing is as follows:

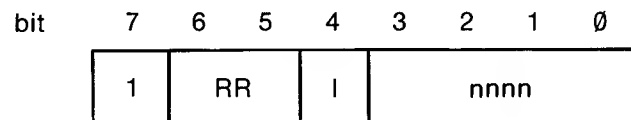
bit	7	6	5	4	3	2	1	0
	0	RR		OFFSET				



where:           RR = 00 if X register  
                       = 01 if Y register  
                       = 10 if U register  
                       = 11 if S register

OFFSET = 5-bit two's complement

- b. Complex format indexing: all indexed addressing modes that do not fall under the simple format indexing category use the complex format for indexing. The postbyte code form for complex format indexing is as follows:



where:           RR = 00 if X or PC register  
                       = 01 if Y register  
                       = 10 if U register  
                       = 11 if S register

and:             I = 0 if non-indirect  
                       = 1 if indirect

and:            nnnn    =    0000 Single auto increment (R+)  
                      =    0001 Double auto increment (R++)  
                      =    0010 Single auto decrement (-R)  
                      =    0011 Double auto decrement (- -R)  
                      =    0100 Zero offset value or no offset  
                      =    0101 Register B is offset (B,R)  
                      =    0110 Register A is offset (A,R)  
                      =    1000 8-bit offset  
                      =    1001 16-bit offset  
                      =    1011 Register D is offset (D,R)  
                      =    1100 8-bit offset with PC register  
                      =    1101 16-bit offset with PC register  
                      =    1111 Extended indirect

**Zero Offset Indexed Addressing.** In this mode, the selected index register contains the address of the data to be used by the instruction.

**Examples:**

LDA     ,U

LDA     0,S

**Constant Offset Indexed Addressing.** In this mode, a two's complement offset and the content of the selected index register are added to form the address of the operand. The content of the index register is unchanged by the addition. There are three ranges of offset:

- a. 5-bit Offset (–16 to +15) - the two's complement 5-bit offset is included in the postbyte code. Bit 4 of the postbyte code is used as the sign bit. Bits 0 through 3 are used to designate the constant offset.
- b. 8-bit Offset (–128 to +127) - the two's complement 8-bit offset is contained in a single byte following the postbyte code.
- c. 16-bit Offset (–32768 to +32767) - the two's complement 16-bit offset is contained in the two bytes following the postbyte code.

**Examples:**

```
LDA    10,U  
  
LDU    SAM,X
```

**Accumulator Offset Indexed Addressing.** This mode of addressing is similar to the constant offset mode of indexed addressing except that the two's complement value in one of the accumulators (A, B, or D) and the content of the specified index register are added to form the address of the operand. The contents of both registers are unchanged by this addition.

**Examples:**

```
LDA    B,X  
  
LDX    D,X
```

**Auto Increment/Decrement Indexed Addressing.** In the auto increment addressing mode, the specified index register contains the address of the operand. Then, after the register is used, it will be incremented by one or two. In the auto decrement mode, the specified index register will be decremented before being used for the address of the data. No offset is permitted in this mode of addressing.

**Examples:**

```
LDA    ,Y+
```

```
LDB    ,-X
```

**Extended Indirect Addressing.** In extended indirect addressing, two bytes following the postbyte code of an indexed addressing instruction are used as a pointer to consecutive locations in memory that contains the new address.

**Examples:**

```
LDA    [SAM]
```

```
LDX    [0F10H]
```

### **Indexed Indirect Addressing (MC6809 only)**

All of the indexed addressing modes with the exception of auto increment/decrement by one or a 5-bit offset may specify an additional level of indirection (refer to table 1-3). In indexed indirect addressing, the address will be contained in the location specified by the content of the index register plus any offset.

**Examples:**

```
LDA    [,Y]
```

```
LDA    [B,X]
```

**Bit Set/Clear Addressing (MC6805 only)**

This mode of addressing applies to instructions which can set or clear any bit on page zero. Page zero consists of all RAM space, I/O registers, and 128 bytes of ROM. The lower three bits in the opcode specify the bit to be set or cleared while the byte following the opcode specifies the address in page zero.

**Bit Test and Branch (MC6805 only)**

This mode of addressing applies to instructions which can test any bit in the first 256 locations (00H through 0FFH) and branch to any location relative to the PC. The byte to be tested will be addressed by the byte following the opcode. The individual bit within that byte is addressed by the lower three bits of the opcode. The third byte is the relative address to be added to the program counter if the branch condition is met. These instructions are three-byte operations. The value of the bit tested will be written to the carry bit in the condition code register.

## Condition Codes

The condition code register contains codes that are relevant to the execution of instructions. The register is actually a group of one-bit registers that contain the following information:

Condition Code	Definition	MC6805 Bit No.	MC6809 Bit No.
C	carry-borrow	0	0
V	overflow		1
Z	zero	1	2
N	negative	2	3
I	IRQ interrupt mask	3	4
H	half-carry	4	5
F	FIRQ interrupt mask		6
E	Entire flag		7

The effect of each instruction on the condition codes is indicated in tables 5-1 and 5-2, Chapter 5. A brief description of each condition code is given in the following paragraphs.

## **Carry/Borrow (C) Register**

The carry-borrow register operates like an extension to the accumulator. In an arithmetic addition operation, the final sum may be 9 bits. If this occurs, the carry-borrow code is set ( $C=1$ ) to indicate a carry. If there was no carry, the C register will be reset ( $C=0$ ). For the arithmetic subtraction operation, the carry-borrow code represents a borrow condition. The condition code, when set ( $C=1$ ), indicates that a borrow condition occurred; when reset ( $C=0$ ), it indicates that there was no borrow.

## **Overflow (V) Register (MC6809 only)**

The overflow condition code register will be set ( $V=1$ ) when a two's complement overflow occurs from an arithmetic operation. If no overflow occurs, the register will be reset ( $V=0$ ).

## **Zero (Z) Register**

Immediately after an operation, the zero-detect circuit will look at the result. If all zeros are detected, the zero register will be set ( $Z=1$ ); otherwise, the zero register will be reset ( $Z=0$ ).

## **Negative (N) Register**

Negative numbers are expressed in the two's complement form with bit 7 indicating the negative quality. Bit 7 will be a 1 if the two's complement was negative. Immediately after an operation, the negative register will look at bit 7 to determine if the result was negative. If so, the condition code (N) will be set ( $N=1$ ). The condition code will be reset ( $N=0$ ) if bit 7 was zero, indicating that the two's complement number represented by the result was zero or positive.

## **IRQ Interrupt Mask (I) Register**

The interrupt mask code is set ( $I=1$ ) to prevent the microprocessor from servicing interrupts on the IRQ line. Interrupt requests from any peripheral device will be ignored by the microprocessor until the interrupt mask code is reset ( $I=0$ ).

## **Half Carry (H) Register**

The half carry code will be set ( $H=1$ ) during execution of an ADC or ADD instruction if there was a carry from bit position 3 to bit position 4. The half carry code will be reset ( $H=0$ ) during these instructions if there was no carry.

The half carry code will be undefined after all subtract operations and should not be used in following operations.

**FIRQ Interrupt Mask (F) Register (MC6809 only)**

The microprocessor will not recognize interrupts from the FIRQ line if this bit is set (F=1).

**Entire Flag (E) Register (MC6809 only)**

The entire flag, when set (E=1), indicates that all the registers were moved to the stack (as opposed to only the program counter and condition code registers). The E code bit is used on a return from an interrupt to indicate the extent of unstacking that is required.

Table 1-1. Instruction Addressing Modes - MC6805

I N S T R U C T I O N	I N H E R E N T	I M M E D I A T E	D I R E C T	E X T E N D E D	I N D E X E D	R E L A T I V E	I N S T R U C T I O N	I N H E R E N T	I M M E D I A T E	D I R E C T	E X T E N D E D	I N D E X E D	R E L A T I V E
ADC		X	X	X	X		CLR	X		X			X
ADD		X	X	X	X		CMP		X	X	X		X
AND		X	X	X	X		COM	X		X			X
ASL	X		X		X		CPX		X	X	X		X
ASR	X		X		X		DEC	X		X			X
BCC						X	EOR		X	X	X		X
*BCLR							INC	X		X			X
BCS						X	JMP			X	X		X
BEQ						X	JSR			X	X		X
BHCC						X	LDA		X	X	X		X
BHCS						X	LDX		X	X	X		X
BHI						X	LSL	X		X			X
BHS						X	LSR	X		X			X
BIH						X	NEG	X		X			X
BIL						X	NOP	X					
BIT		X	X	X	X		ORA		X	X	X		X
BLO						X	ROL	X		X			X
							ROR	X		X			X
BLS						X	RSP	X					
BMC						X	RTI	X					
BMI						X	RTS	X					
BMS						X	SBC		X	X	X		X
BNE						X	SEC	X					
BPL						X	SEI	X					
BRA						X	STA			X	X		X
BRN						X	STX			X	X		X
**BRCLR							SUB		X	X	X		X
**BRSET							SWI	X					
*BSET							TAX	X					
BSR						X	TST	X		X			X
CLC						X	TXA	X					
CLI						X							

**NOTE:** (\*) - Bit Set/Clear mode of addressing.  
(\*\*) - Bit Test and Branch mode of addressing.



Table 1-2. Instruction Addressing Modes - MC6809

IN STRUC TION	IN HER ENT	IM ME DI ATE	DI RECT	EX TE N DE D	EX T IN DR	IN DE X E D	IN DR	RE LA T IVE	RE L IN DR
ABX	X								
ADC		X	X	X	X	X	X	X	X
ADD		X	X	X	X	X	X	X	X
AND		X	X	X	X	X	X	X	X
ASL			X	X	X	X	X	X	X
ASR			X	X	X	X	X	X	X
BCC								X	
BCS								X	
BEQ								X	
BGE								X	
BGT								X	
BHI								X	
BHS								X	
BIT		X	X	X	X	X	X	X	X
BLE								X	
BLO								X	
BLS								X	
BLT								X	
BMI								X	
BNE								X	
BPL								X	
BRA								X	
BRN								X	
BSR								X	
BVC								X	
BVS								X	
CLR			X	X	X	X	X	X	X
CMP		X	X	X	X	X	X	X	X
COM			X	X	X	X	X	X	X
CWAI		X							
DAA	X								
DEC				X	X	X	X	X	X
EOR		X	X	X	X	X	X	X	X
EXG	X								
INC		X	X	X	X	X	X	X	X
JMP			X	X	X	X	X	X	X
JSR			X	X	X	X	X	X	X
LD_		X	X	X	X	X	X	X	X
LEA					X	X	X	X	X
LSL			X	X	X	X	X	X	X
LSR			X	X	X	X	X	X	X
MUL	X								
NEG			X	X	X	X	X	X	X
NOP	X								
ORA		X	X	X	X	X	X	X	X
ORCC		X							
PSHS	X								
PSHU	X								
PULS	X								
PULU	X								
ROL			X	X	X	X	X	X	X
ROR			X	X	X	X	X	X	X
RTI	X								
RTS	X								
SBC		X	X	X	X	X	X	X	X
SEX	X								
ST_			X	X	X	X	X	X	X
SUB		X	X	X	X	X	X	X	X
SWI	X								
SYNC	X								
TRF	X								
TST			X	X	X	X	X	X	X

Table 1-3. Indexed Addressing Modes (MC6809 only)

Mode	Type	Non-Indirect		Indirect	
		Format	Postbyte Op Code	Format	Postbyte Op Code
Constant Offset From R (Signed)	No Offset	,R	1RR00100	[,R]	1RR10100
	5-Bit Offset	n,R	0RRnnnnn	defaults to 8-bit	
	8-Bit Offset	n,R	1RR01000	[n,R]	1RR11000
	16-Bit Offset	n,R	1RR01001	[n,R]	1RR11000
Accm Offset From R (Signed)	A - Reg Offset	A,R	1RR00110	[A,R]	1RR10110
	B - Reg Offset	B,R	1RR00101	[B,R]	1RR10101
	D - Reg Offset	D,R	1RR01011	[D,R]	1RR11011
Auto Incr/ Decr R	Incr by 1	,R+	1RR00000	not allowed	
	Incr by 2	,R++	1RR00001	[,R++]	1RR10001
	Decr by 1	,-R	1RR00010	not allowed	
	Decr by 2	,- -R	1RR00011	[,- -R]	1RR10011
Constant Offset From R	8-Bit Offset	n,PCR	1dd01100	[n,PCR]	1dd11100
	16-Bit Offset	n,PCR	1dd01101	[n,PCR]	1dd11101
Extended Indirect	16-Bit Addr			[n]	10011111

where: R = X, Y, U, or S

and where: X bit code = 00  
Y bit code = 01  
U bit code = 10  
S bit code = 11  
d = don't care  
n = offset

## Operand Rules and Conventions

### Types of Information

There are four types of data that may be needed as items in the operand field:

- a. Register Information - operands may reference directly data contained in the processor registers such as the stack, register A, or the index register.

**Example:**

STA	SAM	;MOVE CONTENTS OF
		;REGISTER A TO SAM

- b. Index Register Information - operands may reference directly data contained in the index register.

**Example:**

LDX	0100H	;LOAD INDEX REGISTER
		;FROM MEMORY

- c. Immediate Data - operands may contain immediate data. The required value is inserted directly into the operand field. The value may be in the form of numbers, an expression to be evaluated at assembly time, a symbol, or an ASCII constant enclosed in quotation marks.

**Examples:**

LDA	#0FFH	;LOAD "FF" HEX INTO ;REGISTER A
LDA	#"A"	;LOAD VALUE OF ASCII ;CONSTANT A (01000001B) ;INTO REGISTER A

- d. 16-bit Memory Address - operands may reference a 16-bit absolute memory address within the range of 0 to 65,535 that contains the operand data.

**Example:**

5FFFH

## Additional Operand Information

### Immediate Addressing Indicator

To select the immediate addressing mode, the corresponding operand must be preceded by the pound (#) character. The data following the (#) sign will be assigned one or two bytes of memory, depending on the instruction.

For MC6809 instructions PSHS, PULS, PSHU, and PULU, any register list (A, B, CC, D, DP, PC, S, U, X, or Y) may be used with the following exceptions:

- Register S cannot be specified with a PSHS or PULS instruction.
- Register U cannot be specified with a PSHU or PULU instruction.

For MC6809 instructions EXG and TFR exactly two registers must be specified in the register list. The following restrictions hold for the two instructions:

- EXG instruction - the two designated registers must be of the same size.
- TFR instruction - the two designated registers must be of the same size, or the first register listed must be a 16-bit register and the second register listed must be an 8-bit register.

## Indexed Addressing Indicator

With this mode of addressing, the numerical address is variable and dependent on the content of one of the MC6809 index registers (S, U, X, or Y) or the MC6805 index register (X). The address is obtained during instruction execution by adding the value of the operand expression to the current content of the index register.

### Example:

```
ADC          10,X
```

## Direct - Extended Addressing Mode Default

For those instructions that can use both direct and extended modes, the assembler defaults to extended for externals, relocatables, and forward references. The direct mode is used when addresses are in the 0 to FFH range. The default function can be overridden by using the DIRECT pseudo instruction. Once the direct pseudo is inserted in a source program, the direct mode of addressing will be in effect until cancelled by an EXTEND pseudo instruction.

While in the direct mode of addressing (MC6809), individual source statements may be assigned the extended mode of addressing by prefixing the operand with a greater-than (>) character.

### Example:

```
CLR          >0F10H
```

While in the extended mode of addressing (EXTEND pseudo in effect), individual source statements (MC6809) may be assigned the direct mode of addressing by prefixing the operand with a less-than (<) character.

### Example:

```
ADDA         <0FH
```

## Location Counter Indicator

The dollar symbol (\$) refers to the current location of the program counter. The program counter contains the address where the current instruction or data statement is being assembled.

**Example:**

JUMP	JMP	+\$3	;JUMP TO ADDRESS ;3 BYTES BEYOND ;FIRST BYTE OF THIS ;INSTRUCTION
------	-----	------	--

## **Operand Expressions**

The operand field may contain an expression consisting of one or more terms acted on by the expression operators listed in Chapter 2 of the Assembler/Linker Reference Manual. A term may be either a symbol, a string constant, a numeric constant, or an expression. The assembler reduces the entire expression to a single value.

# Chapter **3**

## **Special Pseudo Instructions**

### **Introduction**

This chapter supplements Chapter 3 in the HP Model 64000 Assembler/Linker Reference Manual. It lists and defines in detail those assembler instructions that are applicable to the 6805 microcomputer and 6809 microprocessor.

## Declare Symbols Relocatable and on Base Page

### SYNTAX:

Label	Operation	Operand	Comment
	BASE_SEG		
	BASE_END		

The BASE\_SEG and BASE\_END pseudo instructions alert the assembler for symbols that will be on base page although they are relocatable.

BASE\_SEG only affects labels defined by pseudo instructions FCB, FDB, and RMB.

### Example:

Label	Operation	Operand	Comment
	DATA		
	BASE_SEG		
JULY	FCB	0	;JULY is DATA ;relocatable and flagged ;as base page.
JUNE	RMB	12	
	BASE_END		;Turns off base page ;flag.
	PROG		
	LDAA	JULY	;Generates base page ;reference. Linker
	LDAA	JUNE	;checks for errors. ;Labels must be defined ;before using or they ;will not be flagged as ;base page.
	LDAA	AUGUST	;This will not be on ;base page, since it ;is defined out of the ;BASE_SEG range.
AUGUST	FCB	0	



## Declare Symbols External and on Base Page

### SYNTAX:

Label	Operation	Operand	Comment
[Name]	BEXT	operand[, operand, . . . ]	

The BEXT pseudo instruction declares expression as external and on base page. The linker checks for range errors.

### Example:

BEXT	SAM	;SAM is external and on ;base page.
EXT	CHARLIE	;CHARLIE is external.
LDAA	SAM	;Generates base page ;reference.
LDAA	CHARLIE	;Assembler generates ;extended addressing ;unless told to put ;on base page.

**Block Storage of Zeros**

## SYNTAX:

Label	Operation	Operand	Comment
[Name]	BSZ	expression	

The BSZ pseudo instruction allocates a block of bytes. Each byte has an initial value of zero. Expression determines the number of bytes allocated.

An error will be generated if Expression has a value of zero, or contains symbols that are undefined, external references, or forward references.

**Example:**

Label	Operation	Operand	Comment
	BSZ	10	;Generates 10 bytes of ;zeros.

## DIRECT

### Direct Addressing Mode

#### SYNTAX:

Label	Operation	Operand	Comment
	DIRECT		

Some microprocessor instructions can use either the direct or the extended mode of addressing. Unless otherwise instructed, the assembler defaults to extended addressing. To cancel this default condition, insert the **DIRECT** pseudo instruction into the source program.

## EXTEND

### Extended Addressing Mode

#### SYNTAX:

Label	Operation	Operand	Comment
	EXTEND		

The **EXTEND** pseudo instruction selects the extended mode of addressing. To cancel the **EXTEND** instruction, insert the **DIRECT** pseudo instruction into the source program.

**Form Constant Byte****SYNTAX:**

<b>Label</b>	<b>Operation</b>	<b>Operand</b>	<b>Comment</b>
[Name]	FCB	expression	

The FCB pseudo instruction will store data in consecutive memory locations starting with the current setting of the program counter. The operand field may contain symbols or expressions that evaluate to one byte (8 bits) numbers in the range 0 through 255.

The label name is optional. If the label name is present, it is assigned the starting value of the program counter, and will reference the first byte stored by the FCB instruction.

**Example:**

<b>Label</b>	<b>Operation</b>	<b>Operand</b>	<b>Comment</b>
SAM	FCB	CHARLIE+05H	

## Form Constant Character String

### SYNTAX:

Label	Operation	Operand	Comment
[Name]	FCC	number, string expression	
[Name]	FCC or	string expression	

The FCC pseudo instruction stores ASCII strings into consecutive bytes of memory. Any printable ASCII character can be included in the string. This pseudo has two formats. In the first format, Number is a decimal constant, which specifies the number of characters contained in string expression. If Number exceeds the characters in String Expression, spaces will be inserted to fill the remainder of the string.

In the second format, FCC specifies the string, which can be any printable ASCII character within quotation marks (" . "), apostrophe marks (' . '), or carets (^ . ^).

### Examples:

Label	Operation	Operand	Comment
	FCC	10, "TEXT"	;Generates TEXT in ASCII ;followed by 6 blanks.
	FCC	"TEXT"	;Only generates TEXT.

**Form Double Byte**

## SYNTAX:

Label	Operation	Operand	Comment
[Name]	FDB	expression list	

The FDB pseudo instruction will store each 16-bit value from the expression list as an address. The values are stored in memory starting at the current setting of the program counter.

Expressions evaluate to one-word (16 bits) numbers, typically addresses. If an expression evaluates to a single byte, it is assumed to be the low order byte of a 16-bit word where the high order byte is all zeros.

If the label name is present, it is assigned the starting address of the program counter, and thus will reference the first byte stored by the FDB instruction.

**Example:**

Label	Operation	Operand	Comment
SAM	FDB	0B123H	

## Define Register List

### SYNTAX:

Label	Operation	Operand	Comment
Name	REG	register list	

The REG pseudo instruction assigns the value of Register List to Name. Name cannot be redefined elsewhere in the program. Register List must be in the following form:

R1,[R2,...,R<sub>n</sub>] where R1 to R<sub>n</sub> are symbols A,B,CC,D,DP,PC,S,  
U,X, or Y. Both S and U cannot be used at the same time.  
Register D is the same as registers A and B. A register can only  
be specified once with REG.

After REG is defined, Name should only be used with PSHU, PULU, PSHS, and PULS. A "U" in Register List should not be used with PSHU or PULU. An "S" in Register List should not be used with PSHS or PULS.

### Example:

Label	Operation	Operand	Comment
ABREG	REG	A,B	
	PSHS	# ABREG	

**Reserve Memory Byte****SYNTAX:**

<b>Label</b>	<b>Operation</b>	<b>Operand</b>	<b>Comment</b>
[Name]	RMB	expression	

The RMB pseudo instruction may be used to define a block of memory space. The value of the expression in the operand field specifies the number of bytes to be reserved.

Any symbol appearing in the operand field must be predefined. If the value of the operand expression is zero, no memory is reserved; however, if the optional label name is present, it will be assigned the current value of the program counter.

The RMB instruction reserves space in memory by incrementing the program counter by the value in the operand expression.

**Example:**

<b>Label</b>	<b>Operation</b>	<b>Operand</b>	<b>Comment</b>
SAM	RMB	15	;RESERVE 15 ;BYTES FOR SAM ;ROUTINE



## Set Symbol to a Value

### SYNTAX:

Label	Operation	Operand	Comment
Name	SET	expression	

The SET pseudo instruction assigns the value of Expression to Name. The value of Name can be changed later in the program with another SET instruction.

### Example:

Label	Operation	Operand	Comment
SAM	SET	15	;SAM has a value of 15.

(6809 only)

## Set Direct Page Pseudo Register

### SYNTAX:

Label	Operation	Operand	Comment
[Name]	SETDP	expression	

The SETDP pseudo instruction assigns the value of the least significant byte in Expression to the direct page pseudo register at assembly time. SETDP can be used as often as required in an assembly; the value of the direct page pseudo register will change each time. No external references, forward references, or undefined symbols are allowed in Expression and it must be an absolute expression.

### Example:

Label	Operation	Operand	Comment
	SETDP	30H	;The direct page pseudo ;register is set to 30H ;and absolute addresses ;in range 3000H-3000FFH ;are assembled with the ;direct addressing mode.

# Assembler Output Listing

## General

The assembler processes source program modules and produces an output that consists of a source program listing, a relocatable object file, and a symbol cross-reference list. Errors detected by the assembler will be noted in the output listing as error messages.

## Input/Output Files

### Source Input File

The input to the assembler is a source file that is created through the editor. It consists of the following:

Example	Description
"6809"	- Assembler directive.
Source Code	- Source statements consisting of:
.	Assembler Pseudos - refer to Chapter 3 (Assembler/Linker Reference Manual)
.	Assembler Instructions - refer to Chapter 5, this Supplement
.	
.	
.	

### Assembler Output Files

The assembler produces relocatable object modules that are stored under the same name as the source file but in a format that can be processed by the linker. If an object file does not exist at assembly time, the assembler will create one. If an object file does exist, the assembler will replace it.

**List File.** The list file is a formatted file that is output to a line printer. It can also be stored in a file or applied to the system CRT display. The listing may include the following:

- a. Source statements with object code.
- b. Error messages.
- c. Summary of errors with a description list.
- d. Symbol cross-reference list.

**Symbol Cross-reference List.** All symbols are cross-referenced except local macro labels and parameters. A cross-reference listing contains the following:

- a. Alphabetical list of program symbols.
- b. Line numbers where symbols are defined.
- c. All references (by line numbers) to symbols in the program.

## Output Listing

An example of an assembler output listing is given in table 4-2, using the source program example listed in table 4-1. To illustrate an assembler output listing that contains error messages refer to table 4-3.

### NOTE

---

The source program example was not written as a specific program. It merely lists a group of mnemonics to present a formatted example.

---

Table 4-1. Source Program Format Example

"6809" LIST XREF		
KYBD9	GLB	KYBD9
	EXT	DSPL9
	DEC	[10,X]
	DECA	
	DECB	
	EXG	X,Y
	INC	[10,X]
	JMP	DSPL9
	CWAI	#0FH
	JSR	MIKE
	INCA	
	INCB	
	LBCC	1FFDH
	LBGE	1FEEH
	LDA	#0FFH
	LDB	#00H
	LDS	#0500H
	LDU	#0010H
	LEAS	10,U
	LEAX	10,Y
MIKE	LSLA	
	LSLB	
	MUL	
	NEG	10,U
	NOP	
	ORA	15H
	ORB	20H
	ORCC	#0FH
	PSHS	X,Y
	PSHU	S,X
	ROLA	
	ROLB	
	JSR	DSPL9
	END	

Table 4-2. Assembler Output Listing

FILE: PGM09E:		HEWLETT-PACKARD: MOTOROLA 6809 ASSEMBLER			
LINE	LOC	CODE	ADDR	SOURCE STATEMENT	
1				"6809" LIST XREF	
2				GLB	KYBD9
3				EXT	DSPL9
4	0000	6A	98	DEC	[10,X]
5	0003	4A		DECA	
6	0004	5A		DECB	
7	0005	1E	12	EXG	X,Y
8	0007	6C	98	INC	[10,X]
9	000A	0E	00	JMP	DSPL9
10	000C	3C	0F	CWAI	#0FH
11	000E	9D	2E	JSR	MIKE
12	0010	4C		INCA	
13	0011	5C		INCB	
14	0012	1024	1FFD	LBCC	1FFDH
15	0016	102C	1FEE	LBGE	1FEEH
16	001A	86	FF	LDA	#0FFH
17	001C	C6	00	LDB	#00H
18	001E	10CE	0500	LDS	#0500H
19	0022	CE	0010	LDU	#0010H
20	0025	32	4A	LEAS	10,U
21	0027	30	2A	LEAX	10,Y
22	0029	48		LSLA	
23	002A	58		LSLB	
24	002B	3D		MUL	
25	002C	60	4A	NEG	10,U
26	002E	12		NOP	
27	002F	9A	15	ORA	15H
28	0031	DA	20	ORB	20H
29	0033	1A	0F	ORCC	#0FH
30	0035	34	30	PSHS	X,Y
31	0037	36	50	PSHU	S,X
32	0039	49		ROLA	
33	003A	59		ROLB	
34	003B	9D	00	JSR	DSPL9
35				END	
Errors= 0					

Table 4-2. Assembler Output Listing (Cont'd)

FILE: PGM09E:		CROSS REFERENCE TABLE		PAGE 2
LINE#	SYMBOL	TYPE	REFERENCES	
3	DSPL9	E	9	
4	KYBD9	P	2	
26	MIKE	P		
	S	R	31	
	U	R	20,25	
	X	R	4,7,8,30,31	
	Y	R	7,21,30	

**NOTE:** In the cross-reference table, the letter listed under the TYPE column has the following definition:

A = Absolute  
C = Common (COMN)  
D = Data (DATA)  
E = External  
M = Multiple Defined  
P = Program (PROG)  
R = Predefined Register  
U = Undefined

Table 4-3. Assembler Output with Errors

FILE: PGM09E:		HEWLETT-PACKARD: MOTOROLA 6809 ASSEMBLER			
LINE	LOC	CODE	ADDR	SOURCE STATEMENT	
1				"6809" LIST XREF	
2				GLB	KYBD9
3				EXT	DSPL9
4	0000	6A	98	KYBD9	DEC [10,X]
5	0003	4A		DECA	
6	0004	5A		DECB	
7				EXGF	X,Y
				^	
ERROR-UO					
8	0007	6C		INC	[10,X]
9	000A	0E	00	JMP	DSPL9
10				CWAI	\$+1
				^	
ERROR-IO,see line 7					
11	000E	9D	2E	JSR	MIKE
12	0010	4C		INCA	
13	0011	5C		INCB	
14	0012	1024	1FFD	LBCC	1FFDH
15	0016	102C	1FEE	LBGE	1FEEH
16	001A	86	00	LDA	#FFH
				^	
ERROR-US,see line 10					
17	001C	C6	00	LDB	#00H
18	001E	10CE	0500	LDS	#0500H
19	0022	CE	0010	LDU	#0010H
20	0025	32	4A	LEAS	10,U
				^	
ERROR-IO, see line 16					
21	0027	30	2A	LEAX	10,Y
22	0029	48		LSLA	
23	002A	58		LSLB	
24	002B	3D		MUL	
25	002C	60	4A	NEG	10,U
26	002E	12		NOP	
27	002F	9A	15	ORA	15H
28	0031	DA	20	ORB	20H
29				ORCC	FFH
ERROR-US,see line 16					
30	0035	34	30	PSHS	X,Y



Table 4-3. Assembler Output with Errors (Cont'd)

LINE	LOC	CODE	ADDR	SOURCE STATEMENT
31	0037	36	50	PSHU S,X
32	0039	49		ROLA
33	003A	59		ROLB
34	003B	9D	00	JSR DSPL9
35				END

Errors= 4, previous error at line 29

US - Undefined Symbol, the indicated symbol is not defined as a label or declared as an  
 US - external.  
 IO - Invalid Operand, invalid or unexpected operand encountered, or operand is missing.  
 UO - Unidentified Opcode, opcode encountered is not defined for this microprocessor.

FILE: PGM09E:		CROSS REFERENCE TABLE	PAGE 2
LINE#	SYMBOL	TYPE	REFERENCES
3	DSPL9	E	9
***	FFH	U	16,29
4	KYBD9	P	2
26	MIKE	P	
	S	R	31
	U	R	20,25
	X	R	4,8,30,31
	Y	R	21,30

**NOTE:** Error messages are inserted immediately following the statement where the error occurs. All error messages (after the first error message) will contain a statement that points to the line number where the previous error occurred. At the end of the source program listing, an error summary statement will be printed. The summary will contain a statement indicating the total number of errors noted, along with a line reference to the previous error. It will also define all error codes listed in the source program listing.

The primary purpose of the error statement that points to the line number where the previous error occurred is to facilitate location of errors. Since some programs may be many pages in length, this feature helps the programmer locate errors quickly (as opposed to thumbing through each page of the program).



## Instruction Set Summary

### General

All MC6805/6809 instructions are summarized in tables 5-1 and 5-2. The tables are arranged in alphabetical order.

Each instruction consists of a mnemonic symbol, object code for each addressing mode, the boolean operation performed, and condition codes affected. The descriptive symbols used in tables 5-1 and 5-2 to represent items in mnemonic definitions are as follows:

Symbol	Description
A	Register A
B	Register B
C or CC	Carry/borrow condition code
CCR	Condition Code Register
D	Register D (Reg A and Reg B)
Dir	Direct addressing mode
E	Entire condition code
EA	Effective address
Ext	Extended addressing mode
F	FIRQ interrupt code
H	Half-carry condition code
I	IRQ interrupt code
Imm	Immediate addressing mode
Ind	Indexed addressing mode
Inh	Inherent addressing mode
M	A memory location (1 byte)

Symbol	Description
n	Condition code not affected
N	Negative condition code
off.	Offset
OP	Operation Code (Hexadecimal)
PC	Program counter
PCH	Program counter (High Byte)
PCL	Program counter (Low Byte)
PCR	Program counter (Relative)
Rel	Relative Address
S	Register S
SP	Stack Pointer
SPH	Stack Pointer (High Byte)
SPL	Stack Pointer (Low Byte)
U	Register U
u	Condition code unknown
V	Overflow condition code
X	Register X
x	Condition code affected
Y	Register Y
Z	Zero condition code
Ø	Bit = 0
1	Bit = 1
•	Boolean AND
<—	Transfer into
<—>	Exchange
+	Arithmetic plus
—	Arithmetic minus
*	Multiplication indicator
=	Equality indicator
( )	Refers to contents of address or register
⊕	Exclusive OR
⊙	Inclusive OR

## Predefined Symbols

The following symbols are reserved. They have special meaning to the assembler and cannot appear as user-defined symbols.

Symbol	Definition
A	Register A
B	Register B
D	Register D
DP	Direct Page Register
S	Register S
U	Register U
X	Register X
Y	Register Y
SP	Stack Pointer
\$	Program Counter
#	Immediate addressing byte follows
[ ]	Addressing indirection

Table 5-1. MC6805 Instruction Set Summary

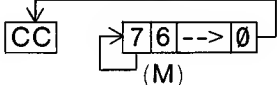
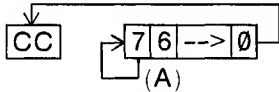
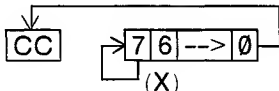
Mnemonic	Object Code	Addr Mode	Operation	Flag Bits	H	I	N	Z	C
					4	3	2	1	0
ADC	A9	Imm	$A \leftarrow (A) + (M) + (C)$		x	n	x	x	x
	B9	Dir							
	C9	Ext							
	F9	Ind (no off.)							
	E9	Ind (8-bit off.)							
	D9	Ind (16-bit off.)							
ADD	AB	Imm	$A \leftarrow (A) + (M)$		x	n	x	x	x
	BB	Dir							
	CB	Ext							
	FB	Ind (no off.)							
	EB	Ind (8-bit off.)							
	DB	Ind (16-bit off.)							
AND	A4	Imm	$A \leftarrow (A) \bullet (M)$		n	n	x	x	n
	B4	Dir							
	C4	Ext							
	F4	Ind (no off.)							
	E4	Ind (8-bit off.)							
	D4	Ind (16-bit off.)							
ASL	38	Dir	$CC \leftarrow [7] \leftarrow [0] \leftarrow 0$ (M)		n	n	x	x	x
	78	Ind (no off.)							
	68	Ind (8-bit off.)							
ASLA	48	Inh	$CC \leftarrow [7] \leftarrow [0] \leftarrow 0$ (A)		n	n	x	x	x
ASLX	58	Inh	$CC \leftarrow [7] \leftarrow [0] \leftarrow 0$ (X)		n	n	x	x	x
ASR	37	Dir			n	n	x	x	x
	77	Ind (no off.)							
	67	Ind (8-bit off.)							

Table 5-1. MC6805 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	Flag	H	I	N	Z	C
				Bits	4	3	2	1	0
ASRA	47	Inh			n	n	x	x	x
ASRX	57	Inh			n	n	x	x	x
BCC	24	Rel	Test for C=0		n	n	n	n	n
BCLR	(11+ 2 x n)	Bit Set/Clear	Clear bit		n	n	n	n	n
BCS	25	Rel	Test for C=1		n	n	n	n	n
BEQ	27	Rel	Test for Z=1		n	n	n	n	n
BHCC	28	Rel	Test for H=0		n	n	n	n	n
BHCS	29	Rel	Test for H=1		n	n	n	n	n
BHI	22	Rel	Test for C ⊙ Z=0		n	n	n	n	n
BHS	24	Rel	Test for C=0		n	n	n	n	n
BIH	2F	Rel	Test for I=1		n	n	n	n	n
BIL	2E	Rel	Test for I=0		n	n	n	n	n
BIT	A5 B5 C5 F5 E5 D5	Imm Dir Ext Ind (no off.) Ind (8-bit off.) Ind (16-bit off.)	(A) • (M)		n	n	x	x	n

**Table 5-1. MC6805 Instruction Set Summary (Cont'd)**

Mnemonic	Object Code	Addr Mode	Operation	Flag Bits	H	I	N	Z	C
					4	3	2	1	0
BLO	25	Rel	Test for C=1		n	n	n	n	n
BLS	23	Rel	Test for $C \odot Z=1$		n	n	n	n	n
BMC	2C	Rel	Test for I=0		n	n	n	n	n
BMI	2B	Rel	Test for N=1		n	n	n	n	n
BMS	2D	Rel	Test for I=1		n	n	n	n	n
BNE	26	Rel	Test for Z=0		n	n	n	n	n
BPL	2A	Rel	Test for N=0		n	n	n	n	n
BRA	20	Rel	Branch always		n	n	n	n	n
BRN	21	Rel	Branch never		n	n	n	n	n
BRCLR	(01+ 2 x n)	Bit Test	Test for bit n=0		n	n	n	n	x
BRSET	(2 x n)	Bit Test	Test for bit n=1		n	n	n	n	x
BSET	(10+ 2 x n)	Bit Set/Clear	Set bit n		n	n	n	n	n
BSR	AD	Rel	Branch to subroutine		n	n	n	n	n
CLC	98	Inh	CC ← 0		n	n	n	n	0
CLI	9A	Inh	I ← 0		n	0	n	n	n
CLR	3F 7F 6F	Dir Ind (no off.) Ind (8-bit off.)	(M) ← 0		n	n	0	1	n



Table 5-1. MC6805 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	Flag Bits	H	I	N	Z	C
					4	3	2	1	0
CLRA	4F	Inh	$(A) \leftarrow 0$		n	n	0	1	n
CLR X	5F	Inh	$(X) \leftarrow 0$		n	n	0	1	n
CMP	A1 B1 C1 F1 E1 D1	Imm Dir Ext Ind (no off.) Ind (8-bit off.) Ind (16-bit off.)	Compare (A), (M)		n	n	x	x	x
COM	33 73 63	Dir Ind (no off.) Ind (8-bit off.)	$(M) \leftarrow \overline{(M)}$		n	n	x	x	1
COMA	43	Inh	$(A) \leftarrow \overline{(A)}$		n	n	x	x	1
COMX	53	Inh	$(X) \leftarrow \overline{(X)}$		n	n	x	x	1
CPX	A3 B3 C3 F3 E3 D3	Imm Dir Ext Ind (no off.) Ind (8-bit off.) Ind (16-bit off.)	Compare (X), (M)		n	n	x	x	x
DEC	3A 7A 6A	Dir Ind (no off.) Ind (8-bit off.)	$(M) \leftarrow (M) - 1$		n	n	x	x	n
DECA	4A	Inh	$(A) \leftarrow (A) - 1$		n	n	x	x	n
DECX	5A	Inh	$(X) \leftarrow (X) - 1$		n	n	x	x	n

**Table 5-1. MC6805 Instruction Set Summary (Cont'd)**

Mnemonic	Object Code	Addr Mode	Operation	Flag Bits	H	I	N	Z	C
					4	3	2	1	0
EOR	A8	Imm	$(A) \leftarrow (A) \oplus (M)$		n	n	x	x	n
	B8	Dir							
	C8	Ext							
	F8	Ind (no off.)							
	E8	Ind (8-bit off.)							
	D8	Ind (16-bit off.)							
INC	3C	Dir	$(M) \leftarrow (M)+1$		n	n	x	x	n
	7C	Ind (no off.)							
	6C	Ind (8-bit off.)							
INCA	4C	Inh	$(A) \leftarrow (A)+1$		n	n	x	x	n
INCX	5C	Inh	$(X) \leftarrow (X)+1$		n	n	x	x	n
JMP	BC	Dir	$PC \leftarrow EA$		n	n	n	n	n
	CC	Ext							
	FC	Ind (no off.)							
	EC	Ind (8-bit off.)							
	DC	Ind (16-bit off.)							
JSR	BD	Dir	$PC \leftarrow EA$		n	n	n	n	n
	CD	Ext							
	FD	Ind (no off.)							
	ED	Ind (8-bit off.)							
	DD	Ind (16-bit off.)							
LDA	A6	Imm	$(A) \leftarrow (M)$		n	n	x	x	n
	B6	Dir							
	C6	Ext							
	F6	Ind (no off.)							
	E6	Ind (8-bit off.)							
	D6	Ind (16-bit off.)							

Table 5-1. MC6805 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	Flag Bits	H	I	N	Z	C
					4	3	2	1	0
LDX	AE	Imm	$(X) \leftarrow (M)$		n	n	x	x	n
	BE	Dir							
	CE	Ext							
	FE	Ind (no off.)							
	EE	Ind (8-bit off.)							
	DE	Ind (16-bit off.)							
LSL	38	Dir	$\boxed{CC} \leftarrow \boxed{7} \leftarrow \boxed{0} \leftarrow \emptyset$ (M)		n	n	x	x	x
	78	Ind (no off.)							
	68	Ind (8-bit off.)							
LSLA	48	Inh	$\boxed{CC} \leftarrow \boxed{7} \leftarrow \boxed{0} \leftarrow \emptyset$ (A)		n	n	x	x	x
LSLX	58	Inh	$\boxed{CC} \leftarrow \boxed{7} \leftarrow \boxed{0} \leftarrow \emptyset$ (X)		n	n	x	x	x
LSR	34	Dir	$\emptyset \rightarrow \boxed{7} \rightarrow \boxed{0} \rightarrow \boxed{CC}$ (M)		n	n	0	x	x
	74	Ind (no off.)							
	64	Ind (8-bit off.)							
LSRA	44	Inh	$\emptyset \rightarrow \boxed{7} \rightarrow \boxed{0} \rightarrow \boxed{CC}$ (A)		n	n	0	x	x
LSRX	54	Inh	$\emptyset \rightarrow \boxed{7} \rightarrow \boxed{0} \rightarrow \boxed{CC}$ (X)		n	n	0	x	x
NEG	30	Dir	$(M) \leftarrow \overline{(M)} + 1$		n	n	x	x	x
	70	Ind (no off.)							
	60	Ind (8-bit off.)							

Table 5-1. MC6805 Instruction Set Summary (Cont'd)

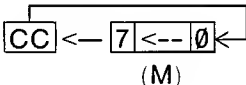
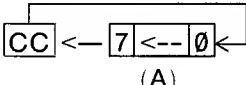
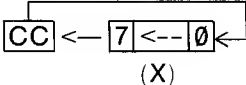
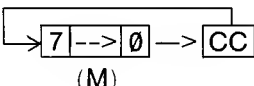
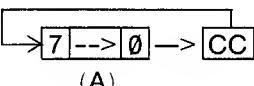
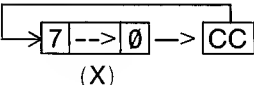
Mnemonic	Object Code	Addr Mode	Operation	Flag	H	I	N	Z	C
				Bits	4	3	2	1	0
NEGA	40	Inh	$(A) \leftarrow (\overline{A})+1$		n	n	x	x	x
NEGX	50	Inh	$(X) \leftarrow (\overline{X})+1$		n	n	x	x	x
NOP	9D	Inh	No operation		n	n	n	n	n
ORA	AA	Imm	$(A) \leftarrow (A) \odot (M)$		n	n	x	x	n
	BA	Dir							
	CA	Ext							
	FA	Ind (no off.)							
	EA	Ind (8-bit off.)							
	DA	Ind (16-bit off.)							
ROL	39	Dir			n	n	x	x	x
	79	Ind (no off.)							
	69	Ind (8-bit off.)							
ROLA	49	Inh			n	n	x	x	x
ROLX	59	Inh			n	n	x	x	x
ROR	36	Dir			n	n	x	x	x
	76	Ind (no off.)							
	66	Ind (8-bit off.)							
RORA	46	Inh			n	n	x	x	x
RORX	56	Inh			n	n	x	x	x

Table 5-1. MC6805 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	Flag Bits	H	I	N	Z	C
					4	3	2	1	0
RSP	9C	Inh	(SP) $\leftarrow$ 07FH		n	n	n	n	n
RTI	80	Inh	Return from interrupt		u	u	u	u	u
RTS	81	Inh	Return from subroutine		n	n	n	n	n
SBC	A2 B2 C2 F2 E2 D2	Imm Dir Ext Ind (no off.) Ind (8-bit off.) Ind (16-bit off.)	(A) $\leftarrow$ (A)–(M)–(CC)		n	n	x	x	x
SEC	99	Inh	(CC) $\leftarrow$ 1		n	n	n	n	1
SEI	9B	Inh	(I) $\leftarrow$ 1		n	1	n	n	n
STA	B7 C7 F7 E7 D7	Dir Ext Ind (no off.) Ind (8-bit off.) Ind (16-bit off.)	(M) $\leftarrow$ (A)		n	n	x	x	n
STX	BF CF FF EF DF	Dir Ext Ind (no off.) Ind (8-bit off.) Ind (16-bit off.)	(M) $\leftarrow$ (X)		n	n	x	x	n
SUB	A0 B0 C0 F0 E0 D0	Imm Dir Ext Ind (no off.) Ind (8-bit off.) Ind (16-bit off.)	(A) $\leftarrow$ (A)–(M)		n	n	x	x	x

**Table 5-1. MC6805 Instruction Set Summary (Cont'd)**

Mnemonic	Object Code	Addr Mode	Operation	Flag	H	I	N	Z	C
				Bits	4	3	2	1	0
SWI	83	Inh	Software Interrupt		n	1	n	n	n
TAX	97	Inh	(X) ← (A)		n	n	n	n	n
TST	3D	Dir	Test (M)		n	n	x	x	n
	7D	Ind (no off.)							
	6D	Ind (8-bit off.)							
TSTA	4D	Inh	Test (A)		n	n	x	x	n
TSTX	5D	Inh	Test (X)		n	n	x	x	n
TXA	9F	Inh	(A) ← (X)		n	n	n	n	n

Table 5-2. MC6809 Instruction Set Summary

Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E	F	H	I	N	Z	V	C
					7	6	5	4	3	2	1	0
ABX	3A	Inh	$X \leftarrow (X) + (B)$		n	n	n	n	n	n	n	n
ADCA	89 99 B9 A9	Imm Dir Ext Ind	$A \leftarrow (A) + (M) + (C)$		n	n	x	n	x	x	x	x
ADCB	C9 D9 F9 E9	Imm Dir Ext Ind	$B \leftarrow (B) + (M) + (C)$		n	n	x	n	x	x	x	x
ADDA	8B 9B BB AB	Imm Dir Ext Ind	$A \leftarrow (A) + (M)$		n	n	x	n	x	x	x	x
ADDB	CB DB FB EB	Imm Dir Ext Ind	$B \leftarrow (B) + (M)$		n	n	x	n	x	x	x	x
ADDD	C3 D3 F3 E3	Imm Dir Ext Ind	$D \leftarrow (D) + (M:M+1)$		n	n	x	n	x	x	x	x
ANDA	84 94 B4 A4	Imm Dir Ext Ind	$A \leftarrow (A) \bullet (M)$		n	n	n	n	x	x	0	n

Table 5-2. MC6809 Instruction Set Summary (Cont'd)

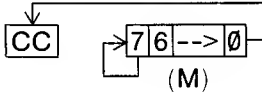
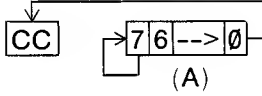
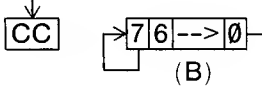
Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E	F	H	I	N	Z	V	C
					7	6	5	4	3	2	1	0
ANDB	C4 D4 F4 E4	Imm Dir Ext Ind	$B \leftarrow (B) \bullet (M)$		n	n	n	n	x	x	0	n
ANDCC	1C	Imm	$CCR \leftarrow (CCR) \bullet \text{data}$		u	u	u	u	u	u	u	1
ASL	08 78 68	Dir Ext Ind	$CC \leftarrow 7 \leftarrow \leftarrow 0 \leftarrow 0$ (M)		n	n	n	n	x	x	x	x
ASLA	48	Inh	$CC \leftarrow 7 \leftarrow \leftarrow 0 \leftarrow 0$ (A)		n	n	n	n	x	x	x	x
ASLB	58	Inh	$CC \leftarrow 7 \leftarrow \leftarrow 0 \leftarrow 0$ (B)		n	n	n	n	x	x	x	x
ASR	07 77 67	Dir Ext Ind			n	n	n	n	x	x	n	x
ASRA	47	Inh			n	n	n	n	x	x	n	x
ASRB	57	Inh			n	n	n	n	x	x	n	x
BCC	24	Rel	Test for $C = 0$		n	n	n	n	n	n	n	n
BCS	25	Rel	Test for $C = 1$		n	n	n	n	n	n	n	n
BEQ	27	Rel	Test for $Z = 1$		n	n	n	n	n	n	n	n



Table 5-2. MC6809 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E	F	H	I	N	Z	V	C
					7	6	5	4	3	2	1	0
BGE	2C	Rel	Test for $N \oplus V = 0$		n	n	n	n	n	n	n	n
BGT	2E	Rel	Test for $Z \odot [N \oplus V] = 0$		n	n	n	n	n	n	n	n
BHI	22	Rel	Test for $C \odot Z = 0$		n	n	n	n	n	n	n	n
BHS	24	Rel	Test for $C = 0$		n	n	n	n	n	n	n	n
BITA	85	Imm	$(A) \bullet (M)$		n	n	n	n	x	x	0	n
	95	Dir										
	B5	Ext										
	A5	Ind										
BITB	C5	Imm	$(B) \bullet (M)$		n	n	n	n	x	x	0	n
	D5	Dir										
	F5	Ext										
	E5	Ind										
BLE	2F	Rel	Test for $Z \odot [N \oplus V] = 1$		n	n	n	n	n	n	n	n
BLO	25	Rel	Test for $C = 1$		n	n	n	n	n	n	n	n
BLS	23	Rel	Test for $C \odot Z = 1$		n	n	n	n	n	n	n	n
BLT	2D	Rel	Test for $N \oplus V = 1$		n	n	n	n	n	n	n	n
BMI	2B	Rel	Test for $N = 1$		n	n	n	n	n	n	n	n
BNE	26	Rel	Test for $Z = 0$		n	n	n	n	n	n	n	n
BPL	2A	Rel	Test for $N = 0$		n	n	n	n	n	n	n	n
BRA	20	Rel	Branch always		n	n	n	n	n	n	n	n

Table 5-2. MC6809 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E	F	H	I	N	Z	V	C
					7	6	5	4	3	2	1	0
BRN	21	Rel	Branch Never		n	n	n	n	n	n	n	n
BSR	8D	Rel	Branch Subroutine		n	n	n	n	n	n	n	n
BVC	28	Rel	Test for V = 0		n	n	n	n	n	n	n	n
BVS	29	Rel	Test for V = 1		n	n	n	n	n	n	n	n
CLR	0F 7F 6F	Dir Ext Ind	(M) ← 0		n	n	n	n	0	1	0	0
CLRA	4F	Inh	(A) ← 0		n	n	n	n	0	1	0	0
CLRB	5F	Inh	(B) ← 0		n	n	n	n	0	1	0	0
CMPA	81 91 B1 A1	Imm Dir Ext Ind	Compare (A), (M)		n	n	n	n	x	x	x	x
CMPB	C1 D1 F1 E1	Imm Dir Ext Ind	Compare (B), (M)		n	n	n	n	x	x	x	x
CMPD	1083 1093 10B3 10A3	Imm Dir Ext Ind	Compare (D), (M:M+1)		n	n	n	n	x	x	x	x
CMPS	118C 119C 11BC 11AC	Imm Dir Ext Ind	Compare (S), (M:M+1)		n	n	n	n	x	x	x	x

Table 5-2. MC6809 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E	F	H	I	N	Z	V	C
					7	6	5	4	3	2	1	0
CMPU	1183	Imm	Compare (U), (M:M+1)		n	n	n	n	x	x	x	x
	1193	Dir										
	11B3	Ext										
	11A3	Ind										
CMPX	8C	Imm	Compare (X), (M:M+1)		n	n	n	n	x	x	x	x
	9C	Dir										
	BC	Ext										
	AC	Ind										
CMPY	108C	Imm	Compare (Y), (M:M+1)		n	n	n	n	x	x	x	x
	109C	Dir										
	10BC	Ext										
	10AC	Ind										
COM	03	Dir	$(M) \leftarrow \overline{(M)}$		n	n	n	n	x	x	0	1
	73	Ext										
	63	Ind										
COMA	43	Inh	$(A) \leftarrow \overline{(A)}$		n	n	n	n	x	x	0	1
COMB	53	Inh	$(B) \leftarrow \overline{(B)}$		n	n	n	n	x	x	0	1
CWAI	3C	Inh	$CCR \leftarrow CCR \odot \text{data}$ ; wait for interrupt		u	u	u	u	u	u	u	1
DAA	19	Inh	Decimal Adj Reg (A)		n	n	n	n	x	x	0	x
DEC	0A	Dir	$(M) \leftarrow (M) - 1$		n	n	n	n	x	x	x	n
	7A	Ext										
	6A	Ind										
DECA	4A	Inh	$(A) \leftarrow (A) - 1$		n	n	n	n	x	x	x	n
DECB	5A	Inh	$(B) \leftarrow (B) - 1$		n	n	n	n	x	x	x	n

Table 5-2. MC6809 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E	F	H	I	N	Z	V	C
					7	6	5	4	3	2	1	0
EORA	88 98 B8 A8	Imm Dir Ext Ind	$(A) \leftarrow (A) \oplus (M)$		n	n	n	n	x	x	0	n
EORB	C8 D8 F8 E8	Imm Dir Ext Ind	$(B) \leftarrow (B) \oplus (M)$		n	n	n	n	x	x	0	n
EXG	1E	Inh	$R1 \leftrightarrow R2$		n	n	n	n	n	n	n	n
INC	0C 7C 6C	Dir Ext Ind	$(M) \leftarrow (M) + 1$		n	n	n	n	x	x	x	n
INCA	4C	Inh	$(A) \leftarrow (A) + 1$		n	n	n	n	x	x	x	n
INCB	5C	Inh	$(B) \leftarrow (B) + 1$		n	n	n	n	x	x	x	n
JMP	0E 7E 6E	Dir Ext Ind	$PC \leftarrow EA$		n	n	n	n	n	n	n	n
JSR	9D  BD AD	Dir  Ext Ind	$(SP) = (SP) - 1,$ $(SP) \text{ PCL};$ $(SP) = (SP) - 1,$ $(SP) \leftarrow PCH;$ $PC \leftarrow EA$		n	n	n	n	n	n	n	n
LBCC	1024	Rel	Test for $C = 0$		n	n	n	n	n	n	n	n
LBCS	1025	Rel	Test for $C = 1$		n	n	n	n	n	n	n	n

Table 5-2. MC6809 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E	F	H	I	N	Z	V	C
					7	6	5	4	3	2	1	0
LBEQ	1027	Rel	Test for $Z = 1$		n	n	n	n	n	n	n	n
LBGE	102C	Rel	Test for $N \oplus V = 0$		n	n	n	n	n	n	n	n
LBGT	102E	Rel	Test for $Z \odot [N \oplus V] = 0$		n	n	n	n	n	n	n	n
LBHI	1022	Rel	Test for $C \odot Z = 0$		n	n	n	n	n	n	n	n
LBHS	1024	Rel	Test for $C = 0$		n	n	n	n	n	n	n	n
LBLE	102F	Rel	Test for $Z \odot [N \oplus V] = 1$		n	n	n	n	n	n	n	n
LBLO	1025	Rel	Test for $C = 1$		n	n	n	n	n	n	n	n
LBLS	1023	Rel	Test for $C \odot Z = 1$		n	n	n	n	n	n	n	n
LBLT	102D	Rel	Test for $N \oplus V = 1$		n	n	n	n	n	n	n	n
LBMI	102B	Rel	Test for $N = 1$		n	n	n	n	n	n	n	n
LBNE	1026	Rel	Test for $Z = 0$		n	n	n	n	n	n	n	n
LBPL	102A	Rel	Test for $N = 0$		n	n	n	n	n	n	n	n
LBRA	16	Rel	Branch always		n	n	n	n	n	n	n	n
LBRN	1021	Rel	Branch never		n	n	n	n	n	n	n	n
LBSR	17	Rel	Branch Subroutine		n	n	n	n	n	n	n	n
LBVC	1028	Rel	Test for $V = 0$		n	n	n	n	n	n	n	n
LBVS	1029	Rel	Test for $V = 1$		n	n	n	n	n	n	n	n

Table 5-2. MC6809 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E	F	H	I	N	Z	V	C
					7	6	5	4	3	2	1	0
LDA	86 96 B6 A6	Imm Dir Ext Ind	(A) ← (M)		n	n	n	n	x	x	0	n
LDB	C6 D6 F6 E6	Imm Dir Ext Ind	(B) ← (M)		n	n	n	n	x	x	0	n
LDD	CC DC FC EC	Imm Dir Ext Ind	(D) ← (M:M+1)		n	n	n	n	x	x	0	n
LDS	10CE 10DE 10FE 10EE	Imm Dir Ext Ind	(S) ← (M:M+1)		n	n	n	n	x	x	0	n
LDU	CE DE FE EE	Imm Dir Ext Ind	(U) ← (M:M+1)		n	n	n	n	x	x	0	n
LDX	8E 9E BE AE	Imm Dir Ext Ind	(X) ← (M:M+1)		n	n	n	n	x	x	0	n
LDY	108E 109E 10BE 10AE	Imm Dir Ext Ind	(Y) ← (M:M+1)		n	n	n	n	x	x	0	n

Table 5-2. MC6809 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E 7	F 6	H 5	I 4	N 3	Z 2	V 1	C 0
LEAS	32	Ind	(S) ← EA		n	n	n	n	n	n	n	n
LEAU	33	Ind	(U) ← EA		n	n	n	n	n	n	n	n
LEAX	30	Ind	(X) ← EA		n	n	n	n	n	n	n	n
LEAY	31	Ind	(Y) ← EA		n	n	n	n	n	n	n	n
LSL	08	Dir	$\boxed{CC} \leftarrow \boxed{7} \leftarrow \boxed{0} \leftarrow \emptyset$ (M)		n	n	n	n	x	x	x	x
	78	Ext										
	68	Ind										
LSLA	48	Inh	$\boxed{CC} \leftarrow \boxed{7} \leftarrow \boxed{0} \leftarrow \emptyset$ (A)		n	n	n	n	x	x	x	x
LSLB	58	Inh	$\boxed{CC} \leftarrow \boxed{7} \leftarrow \boxed{0} \leftarrow \emptyset$ (B)		n	n	n	n	x	x	x	x
LSR	04	Dir	$\emptyset \rightarrow \boxed{7} \rightarrow \boxed{0} \rightarrow \boxed{CC}$ (M)		n	n	n	n	0	x	n	x
	74	Ext										
	64	Ind										
LSRA	44	Inh	$\emptyset \rightarrow \boxed{7} \rightarrow \boxed{0} \rightarrow \boxed{CC}$ (A)		n	n	n	n	0	x	n	x
LSRB	54	Inh	$\emptyset \rightarrow \boxed{7} \rightarrow \boxed{0} \rightarrow \boxed{CC}$ (B)		n	n	n	n	0	x	n	x
MUL	3D	Inh	(D) ← (A)*(B)		n	n	n	n	n	x	n	x
NEG	00	Dir	(M) ← $\overline{(M)} + 1$		n	n	u	n	x	x	x	x
	70	Ext										
	60	Ind										

Table 5-2. MC6809 Instruction Set Summary (Cont'd)

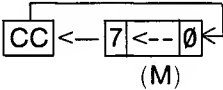
Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E	F	H	I	N	Z	V	C
					7	6	5	4	3	2	1	0
NEGA	40	Inh	$(A) \leftarrow \overline{(A)} + 1$		n	n	u	n	x	x	x	x
NEGB	50	Inh	$(B) \leftarrow \overline{(B)} + 1$		n	n	u	n	x	x	x	x
NOP	12	Inh	No operation (PC) = (PC)+1		n	n	n	n	n	n	n	n
ORA	8A 9A BA AA	Imm Dir Ext Ind	$(A) \leftarrow (A) \odot (M)$		n	n	n	n	x	x	0	n
ORB	CA DA FA EA	Imm Dir Ext Ind	$(B) \leftarrow (B) \odot (M)$		n	n	n	n	x	x	0	n
ORCC	1A	Imm	$(CCR) \leftarrow (CCR) \odot \text{data}$		u	u	u	u	u	u	u	u
PSHS	34	Inh	Push Reg(s) on hardware stack (S)		n	n	n	n	n	n	n	n
PSHU	36	Inh	Push Reg(s) on user stack (U)		n	n	n	n	n	n	n	n
PULS	35	Inh	Pull Reg(s) from hardware stack (S)		u	u	u	u	u	u	u	u
PULU	37	Inh	Pull Reg(s) from user stack (U)		u	u	u	u	u	u	u	u
ROL	09 79 69	Dir Ext Ind			n	n	n	n	x	x	x	x



Table 5-2. MC6809 Instruction Set Summary (Cont'd)

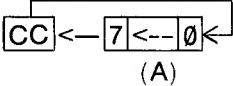
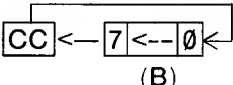
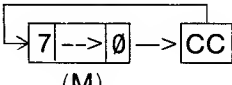
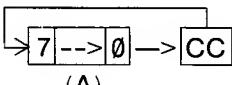
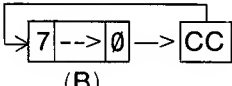
Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E	F	H	I	N	Z	V	C
					7	6	5	4	3	2	1	0
ROLA	49	Inh	 (A)		n	n	n	n	x	x	x	x
ROLB	59	Inh	 (B)		n	n	n	n	x	x	x	x
ROR	06	Dir	 (M)		n	n	n	n	x	x	n	x
	76	Ext										
	66	Ind										
RORA	46	Inh	 (A)		n	n	n	n	x	x	n	x
RORB	56	Inh	 (B)		n	n	n	n	x	x	n	x
RTI	3B	Inh	Return from Interrupt		u	u	u	u	u	u	u	u
RTS	39	Inh	Return from Subroutine		n	n	n	n	n	n	n	n
SBCA	82	Imm	(A) ← (A) - (M) - (CC)		n	n	n	n	x	x	x	x
	92	Dir										
	B2	Ext										
	A2	Ind										
SBCB	C2	Imm	(B) ← (B) - (M) - (CC)		n	n	n	n	x	x	x	x
	D2	Dir										
	F2	Ext										
	E2	Ind										
SEX	1D	Inh	Sign Extended		n	n	n	n	x	x	0	n

Table 5-2. MC6809 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E 7	F 6	H 5	I 4	N 3	Z 2	V 1	C 0
STA	97 B7 A7	Dir Ext Ind	(M) ← (A)		n	n	n	n	x	x	0	n
STB	D7 F7 E7	Dir Ext Ind	(M) ← (B)		n	n	n	n	x	x	0	n
STD	DD FD ED	Dir Ext Ind	(M:M+1) ← (D)		n	n	n	n	x	x	0	n
STS	10DF 10FF 10EF	Dir Ext Ind	(M:M+1) ← (S)		n	n	n	n	x	x	0	n
STU	DF FF EF	Dir Ext Ind	(M:M+1) ← (U)		n	n	n	n	x	x	0	n
STX	9F BF AF	Dir Ext Ind	(M:M+1) ← (X)		n	n	n	n	x	x	0	n
STY	109F 10BF 10AF	Dir Ext Ind	(M:M+1) ← (Y)		n	n	n	n	x	x	0	n
SUBA	80 90 B0 A0	Imm Dir Ext Ind	(A) ← (A) - (M)		n	n	n	n	x	x	x	x

Table 5-2. MC6809 Instruction Set Summary (Cont'd)

Mnemonic	Object Code	Addr Mode	Operation	FLAG BITS	E	F	H	I	N	Z	V	C
					7	6	5	4	3	2	1	0
SUBB	C0	Imm	$(B) \leftarrow (B) - (M)$		n	n	n	n	x	x	x	x
	D0	Dir										
	F0	Ext										
	E0	Ind										
SUBD	83	Imm	$(D) \leftarrow (D) - (M:M+1)$		n	n	n	n	x	x	x	x
	93	Dir										
	B3	Ext										
	A3	Ind										
SWI	3F	Inh	Software Interrupt 1		n	n	n	n	n	n	n	n
SWI2	103F	Inh	Software Interrupt 2		n	n	n	n	n	n	n	n
SWI3	113F	Inh	Software Interrupt 3		n	n	n	n	n	n	n	n
SYNC	13	Inh	Sync to interrupt		n	n	n	n	n	n	n	n
TFR	1F	Inh	$(R2) \leftarrow (R1)$		n	n	n	n	n	n	n	n
TST	0D	Dir	Test (M)		n	n	n	n	x	x	0	n
	7D	Ext										
	6D	Ind										
TSTA	4D	Inh	Test (A)		n	n	n	n	x	x	0	n
TSTB	5D	Inh	Test (B)		n	n	n	n	x	x	0	n





